Hierarchical Text Generation and Planning for Strategic Dialogue

Denis Yarats [*1] **Mike Lewis** [*1]

Abstract

End-to-end models for goal-orientated dialogue are challenging to train, because linguistic and strategic aspects are entangled in latent state vectors. We introduce an approach to learning representations of messages in dialogues by maximizing the likelihood of subsequent sentences and actions, which decouples the semantics of the dialogue utterance from its linguistic realization. We then use these latent sentence representations for hierarchical language generation, planning and reinforcement learning. Experiments show that our approach increases the end-task reward achieved by the model, improves the effectiveness of long-term planning using rollouts, and allows self-play reinforcement learning to improve decision making without diverging from human language. Our hierarchical latent-variable model outperforms previous work both linguistically and strategically.

1. Introduction

Word-by-word approaches to text generation have been successful in many tasks. However, they have limitations in under-constrained generation settings, such as dialogue response or summarization, where models have significant freedom in the semantics of the text to generate. In such cases, models are prone to overly generic responses that may be valid but suboptimal (Li et al., 2015; 2016; Das et al., 2017). Further, such models are uninterpretable and somewhat intellectually dissatisfying because they do not cleanly distinguish between the semantics of language and its surface realization. Entangling form and meaning is problematic for reinforcement learning, where backpropagating caused by semantic decisions can adversely affect the linguistic quality of text (Lewis et al., 2017), and for candidate generation for longterm planning, as linguistically diverse text may lack semantic diversity.

[*]Equal contribution [1]Facebook AI Research, Menlo Park, CA. Correspondence to: Denis Yarats <denisy@fb.com>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

We focus on negotiation dialogues, where the text generated by the model has consequences than can be easily measured. Substitutions of similar words (for example substituting a "one" for a "two") can have a large impact on the end-task reward achieved by a dialogue agent. We use a hierarchical generation approach for a strategic dialogue agent, where the agent first samples a short-term plan in the form of a latent sentence representation. The agent then conditions on this plan during generation, allowing precise and consistent generation of text to achieve a short-term goal. Doing so, we aim to disentangle the concepts of *"what to say"* and *"how to say it"*. To do this, we introduce a method for learning discrete latent representations of sentences based on their effect on the continuation of the dialogue.

Recent work has explored hierarchical generation of dialogue responses, where a latent variable z_t is inferred to maximize the likelihood of a message x_t, given previous messages $x_{0:t-1} \equiv (x_0, \ldots, x_{t-1})$ (Serban et al., 2016a;c; Wen et al., 2017; Cao & Clark, 2017), which has the effect of clustering similar message strings. Our approach differs in that the latent variable z_t is optimized to maximize the likelihood of messages and actions of the continuation of the dialogue, but not the message x_t itself. Hence, z_t learns to represent x_t's effect on the dialogue, but not the words of x_t.The distinction is important because messages with similar words can have very different semantics; and conversely the same meaning can be conveyed with different sentences. We show empirically and through human evaluation that our method leads both to better perplexities and end task rewards, and qualitatively that our representations group sentences that are more semantically coherent but linguistically diverse.

We use our message representations to improve the strategic decision making of our dialogue agent. We improve the model's ability to plan ahead by creating a set of semantically diverse candidate messages by sampling distinct z_t, and then use rollouts to identify the an expected reward for each. We also apply reinforcement learning based on the end-task reward. Previous work has found that RL can adversely effect the fluency of the language generated by the model We instead show that simply fine-tuning the parameters for choosing z_t allows the model to substantially improve its rewards while maintaining human-like language.

Experiments show that our approach to disentangling the form and meaning of sentences leads to agents that use language more fluently and intelligently to achieve their goals.

2. Background

2.1. Natural Language Negotiations

We focus on the negotiation task introduced by Lewis et al. (2017), as it possess both linguistic and reasoning challenges. Lewis et al. collected a corpus of human dialogues on a multi-issue bargaining task, where the agents must divide a collection of items of 3 different types (*books*, *hats* and *balls*) between them. Actions correspond to choosing a particular subset of the items, and agents choose compatible actions if each item is assigned to exactly one agent.

More formally, the agents X and Y are initially given a space $\mathcal{A}$ of possible agreements, and value functions v^X and v^Y, which specify a non-negative reward for each agreement $a \in \mathcal{A}$. Agents cannot directly observe each other's value functions and can only infer it through a dialogue. The agents sequentially exchange turns of natural language x_t, consisting of $n_t + 1$ words $x_t^{0:n_t} \equiv (x_t^0, \ldots, x_t^{n_t})$, until one agent enters a special turn that ends the dialogue. Then, both agents independently enter agreements $a^X, a^Y \in \mathcal{A}$ respectively. If the agreements are compatible, both agents receive a reward based on their action and the value function. If the actions are incompatible, neither agent receives any reward. Training dialogues from an agent's perspective consist of agreement space $\mathcal{A}$, value function v, messages $x_{0:T}$ and agreement a.

2.2. Challenges in Text Generation

We identify a number of challenges for end-to-end text generation for strategic dialogue. These problems have been identified in other text generation settings, but strategic dialogue makes an interesting test case, where decisions have measurable consequences.

- **Lack of semantic diversity**: Multiple samples from a model are often paraphrases of the same intent. This lack of a diversity is a problem if samples are later re-ranked by a long-term planning model.
- **Lack of linguistic diversity**: Neural language models often capture the head of the distribution, providing less varied language than people (Li et al., 2015).
- **Lack of internal coherence**: Messages generated by the model often lack self consistency—for example, *I'll take one hat, and give you all the hats*.
- **Lack of contextual coherence**: Utterances may also lack coherence given the dialogue context so far. For example, Lewis et al. (2017) identify cases where a model starts a message by indicating agreement, but then proposes a counter offer.
- **Entanglement of linguistic and strategic parameters**: End-to-end approaches do not cleanly distinguish between what to say and how to say it. This is problematic as reinforcement learning aiming to improve decision making may adversely affect the quality of the generated language.

We argue that these limitations partly stem from the word-by-word sampling approach to generation, with no explicit plan in advance of generation for what the meaning of the sentence is to be. In §9, we show our hierarchical approach to generation helps with these problems.

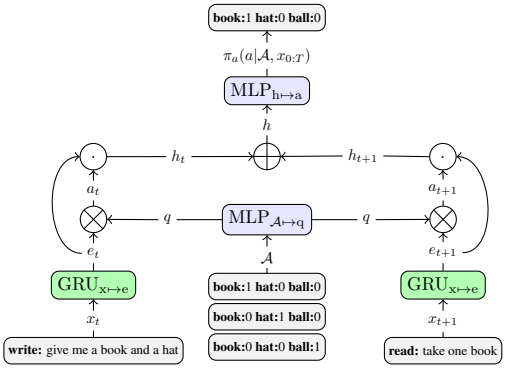


Figure 1: Action classifier $\pi_a(a|\mathcal{A}, x_{0:T})$, which predicts distribution over actions using a GRU with attention.

3. Action Classifier

Initially, we train an *action classifier* $\pi_a(a|\mathcal{A}, x_{0:T})$ (Figure 1) that predicts the final action chosen at the end of the dialogue. This classifier is used in all versions of our model. We implement the action classifier as an RNN with attention (Bahdanau et al., 2014). We first encode the set of possible actions $\mathcal{A}$ as $q = \text{MLP}_{\mathcal{A}\mapsto s}(\mathcal{A})$, and each sentence x_t as $e_t = \text{GRU}_{x\mapsto e}(Ex_t^{0:n_t})$. We then acquire a fixed size representation h by applying the following transformations: $a_t = e_t * q, h_t = e_t \odot a_t, h = \sum_{t=0}^{T} h_t$. Finally, we apply a softmax classifier:

$$\pi_a(a|\mathcal{A}, x_{0:T}) \propto \exp(\text{MLP}_{h\mapsto a}(h))$$

We train this network to minimize the negative log likelihood of an action a given a set of possible actions $\mathcal{A}$ and a dialogue $x_{0:T}$:

$$\mathcal{L} = -\sum_{a,\mathcal{A},x_{0:T}} \log \pi_a(a|\mathcal{A}, x_{0:T})$$

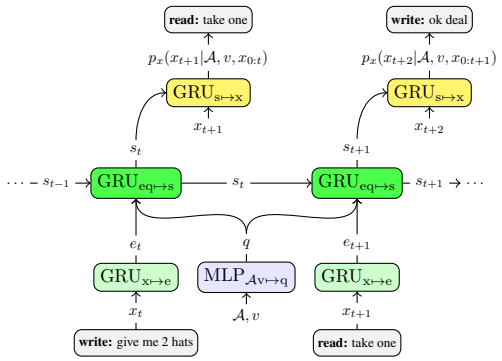


Figure 2: Baseline hierarchical model.

4. Baseline Hierarchical Model

As a baseline, we train a hierarchical encoder-decoder model (Figure 2) to maximize the likelihood of training dialogue sentences, similarly to Serban et al. (2016b). The model contains the action-value encoder to input the action space $\mathcal{A}$ and the value function v as $q = \text{MLP}_{\mathcal{A}v\mapsto q}(\mathcal{A}, v)$; a sentence encoder that embeds individual messages x_t as $e_t = \text{GRU}_{x\mapsto e}(Ex_t^{0:n_t})$; a sentence level encoder that reads sentence embeddings $e_{0:t}$ and the action space encoding q to produce dialogue state $s_t = \text{GRU}_{eq\mapsto s}(e_{0:t}, q)$; and a decoder $\text{GRU}_{s\mapsto x}$ that produces message x_{t+1}, conditioning on s_t:

$$p_x(x_{t+1}^i|\mathcal{A}, v, x_{t+1}^{0:i-1}, x_{0:t}) \propto \exp(E^\top \text{GRU}_{s\mapsto x}(x_{t+1}^{0:i-1}; s_t))$$

$$p_x(x_{t+1}|\mathcal{A}, v, x_{0:t}) = \prod_{i=0}^{n_{t+1}} p_x(x_{t+1}^i|\mathcal{A}, v, x_{t+1}^{0:i-1}, x_{0:t})$$

The encoder and decoder share a word embedding matrix E. We minimize the following loss, over the training set:

$$\mathcal{L} = -\sum_{\mathcal{A},v,x_{0:T}} \sum_{t=0}^{T} \log p_x(x_t|\mathcal{A}, v, x_{0:t-1})$$

5. Learning Latent Message Representations

The central part of our model is a method for encoding messages x_t as discrete latent variables z_t. The goal of this model is to learn message representations that reflect the message's effect on the dialogue, but abstract over semantically equivalent paraphrases. The discrete nature of the latent variables z_t allows us to efficiently make sequential decisions by choosing z_t at each step to govern the outcome of the dialogue. We show that such approach is helpful for planning and reinforcement learning.

Our representation learning model (Figure 3a) has a similar structure to that of §4, except that message embedding e_t is used as input to a stochastic node $p_z(z_t|\mathcal{A}, x_t)$ formed by a softmax with parameters $W_{eq\mapsto z}$ over latent states z_t. We use expectation maximization to learn how to assign messages to clusters to maximize the likelihood of future messages and actions.

After each message x_t, $\text{GRU}_{z\mapsto s}$ is updated with representation z_t to give hidden state s_t. From s_t, we train the model to predict the next message x_{t+1} and an action a_t. In the training dialogues, there is only an action after the final turn x_T; for other turns x_t, we use a soft proxy action by regressing to the distribution over actions predicted by $a_t = \pi_a(a|\mathcal{A}, x_{0:t})$. Therefore, a_t is a distribution over what deal would be agreed if the dialogue stopped after message x_t. This action can be thought of as latent proxy for a traditional annotated dialogue state (Williams et al., 2013). When predicting x_{t+1} and a_t, the model only has access to latent variables $z_{0:t}$, so z_t must contain useful information about the meaning of x_t. We employ a hierarchical RNN, in which message $e_t = \text{GRU}_{x\mapsto e}(Ex_t^{0:n_t})$ and the action space $q = \text{MLP}_{\mathcal{A}\mapsto q}(\mathcal{A})$ encodings are passed through a discrete bottleneck:

$$p_z(z_t|\mathcal{A}, x_t) \propto \exp(W_{eq\mapsto z}[e_t, q])$$

$$s_t = \text{GRU}_{z\mapsto s}(z_{0:t})$$

$$p_x(x_{t+1}^i|\mathcal{A}, x_{t+1}^{0:i-1}, z_{0:t}) \propto \exp(E^\top \text{GRU}_{s\mapsto x}(x_{t+1}^{0:i-1}; s_t))$$

$$p_x(x_{t+1}|\mathcal{A}, z_{0:t}) = \prod_{i=0}^{n_{t+1}} p_x(x_{t+1}^i|\mathcal{A}, x_{t+1}^{0:i-1}, z_{0:t})$$

$$p_a(a_t|\mathcal{A}, z_{0:t}) \propto \exp(\text{MLP}_{s\mapsto a}(s_t))$$

We minimize the following loss over the training set:

$$\mathcal{L} = \sum_{\mathcal{A},x_{0:T}} \sum_{t=0}^{T} -\log p_x(x_{t+1}|\mathcal{A}, z_{0:t}) + \text{D}_{\text{KL}}\left(p_a(a_t|\mathcal{A}, z_{0:t})||\pi_a(a|\mathcal{A}, x_{0:t})\right)$$

We optimize latent variables z using minibatch Viterbi Expectation Maximization (Dempster et al., 1977). For each minibatch, for each timestep t, we compute:

$$z_t^* = \underset{z}{\text{argmax}} \log(p(x_{t+1}, a_t|\mathcal{A}, z, z_{0:t-1})p_z(z|\mathcal{A}, x_t))$$

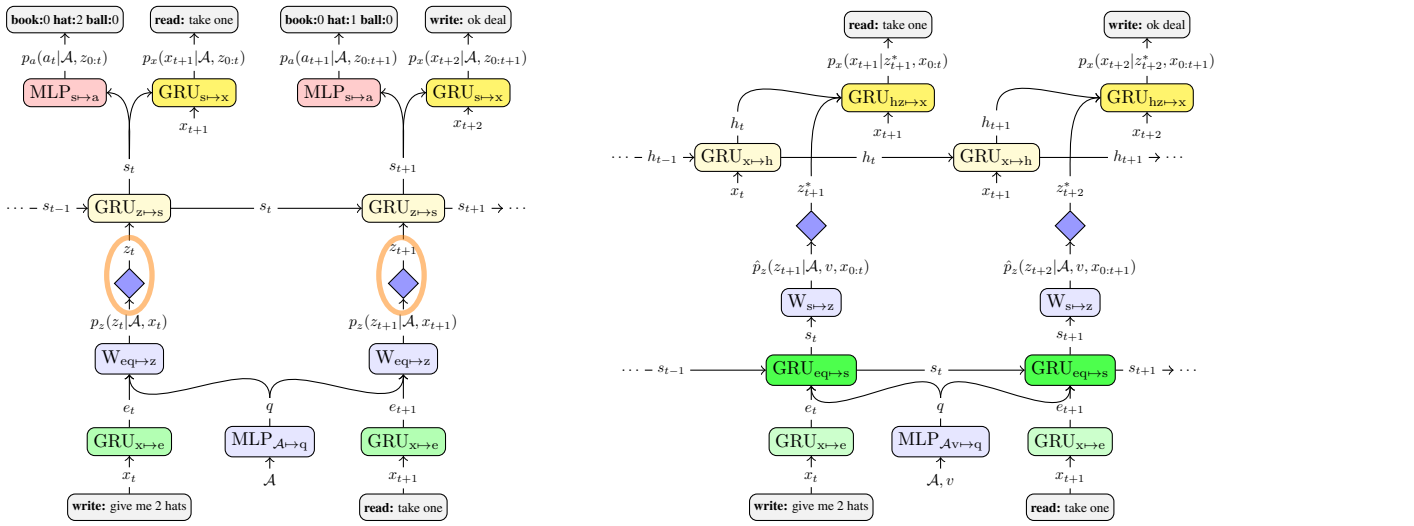
The argmax requires a separate forward pass for each z. We then advance to the next timestep using z_t^* to update $\text{GRU}_{z\mapsto s}$, and finally perform an update maximizing:

$$\sum_{t=0}^{T} \log(p(x_{t+1}, a_t|\mathcal{A}, z_t^*, z_{0:t-1})p_z(z_t^*|\mathcal{A}, x_t))$$

At convergence, we extract message representations z_t^*.

6. Hierarchical Text Generation

We then train a new hierarchical dialogue model (Figure 3b), which uses pre-trained representations z_t^* to predict



(a) Clustering Model

(b) Full Model

Figure 3: We pre-train a model to learn a discrete encoder for sentences, which bottlenecks the message x_t through discrete representation z_t (Figure 3a; §5). This architecture forces z_t to capture the most relevant aspects of x_t for predicting future messages and actions. We then extract the learned discrete representations z_t (marked by orange ellipses) and train our full model (Figure 3b): p_x is trained to translate representations z_t^* into messages x_t (§6.1), and $\hat{p}_z$ is trained to predict a distribution over z_t given the dialogue history (§6.2).

messages x_t. First, we train a recurrent neural network to predict $p_x(x_{t+1}|z^*_{t+1}, x_{0:t})$. p_x learns how to translate the latent variables into fluent text in context. Then, we optimize a model $\hat{p}_z(z_{t+1}|\mathcal{A}, v, x_{0:t})$ to maximize the marginal likelihood of training sentences.

6.1. Conditional Language Model

We train p_x to translate pretrained representation z_t^* and encodings of previous messages $h_t = \text{GRU}_{x\mapsto h}(Ex_{0:t})$ into a message x_t:

$$p_x(x^i_{t+1}|z^*_{t+1}, x_{0:t}, x^{0:i-1}_{t+1}) \propto \exp(E^\top \text{GRU}_{hz\mapsto x}(x^{0:i-1}_{t+1}; z^*_{t+1}, h_t))$$

$$p_x(x_{t+1}|z^*_{t+1}, x_{0:t}) = \prod_{i=0}^{n_{t+1}} p_x(x^i_{t+1}|z^*_{t+1}, x_{0:t}, x^{0:i-1}_{t+1})$$

By minimizing the following loss:

$$\mathcal{L}_x = -\sum_{x_{0:T}} \sum_{t=0}^{T} \log p_x(x_{t+1}|z^*_{t+1}, x_{0:t})$$

Unlike the baseline model, text generation does not condition explicitly on the agent's value function v, or the action space $\mathcal{A}$ – all knowledge of the goals and available actions is bottlenecked through the dialogue state. This restriction forces the text generation to depend strongly on z_t.

6.2. Latent Variable Prediction Model

At test time, z_t^* is not available, as it contains information about the future dialogue. Instead, we train a model $\hat{p}_z$ to predict z_t^* conditioned on the current dialogue context $s_t = \text{GRU}_{e\mapsto s}(e_{0:t}, q)$, where $_t = \text{GRU}_{x\mapsto e}(Ex_t^{0:n_t})$ and $q = \text{MLP}_{\mathcal{A}v\mapsto q}(\mathcal{A}, v)$:

$$\hat{p}_z(z_{t+1}|\mathcal{A}, v, x_{0:t}) \propto \exp(W_{s\mapsto z} s_t)$$

We optimize $\hat{p}_z$ to maximize the marginal likelihood of training messages, without updating p_x. The model learns to reconstruct the distribution over z_t that best explains message x_t.

$$P(x_t|\mathcal{A}, v, x_{0:t-1}) = \sum_z p_x(x_t|z, x_{0:t-1}) \hat{p}_z(z|\mathcal{A}, v, x_{0:t-1})$$

$$\mathcal{L}_z = -\sum_{\mathcal{A},v,x_{0:T}} \sum_{t=0}^{T} \log P(x_t|\mathcal{A}, v, x_{0:t-1})$$

6.3. Decoding

To generate an utterance x_t, the model first samples a predicted plan z_t from $\hat{p}_z$:

$$z_t \sim \hat{p}_z(z|\mathcal{A}, v, x_{0:t-1})$$

The model then sequentially generates tokens x_t^i based on plan z_t and context $x_{0:t}$:

$$x_t^{i+1} \sim p_x(x|z_t, x_{0:t-1}, x_t^{0:i})$$

7. Hierarchial Reinforcement Learning

Lewis et al. (2017) experiment with end-to-end reinforcement learning to fine-tune pre-trained supervised models. The model engages in a dialogue with another model, achieving reward V. This reward is then backpropagated using policy gradients. One challenge is that because model parameters govern both strategic and linguistic aspects of generation, backpropagating errors can adversely affect the quality of the generated language. To avoid divergence from human language, we experiment with fixing all model parameters, except for the parameters of $\hat{p}_z$. This allows reinforcement learning to improve decisions about what to say, without affecting language generation parameters. A similar approach was taken in a different dialogue setting by Wen et al. (2017).

8. Hierarchical Planning

Lewis et al. (2017) propose planning in dialogue using rollouts. First, a set of K unique candidate messages $\{x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(K)}\}$ are sampled from $p_x(x_t|\mathcal{A}, v, x_{0:t-1})$. Then, multiple rollouts of the future dialogue are sampled from the model, and outcomes a are scored according to the value function v, to estimate the expected reward $V(x_t)$:

$$V(x_t) = \mathbb{E}_{x_{t+1:T} \sim p_x, a \sim \pi_a}[r(a, v)\pi_a(a|\mathcal{A}, x_{0:T})] \quad (1)$$

The expectation is approximated with N samples, and the candidate x^* with the highest expected score is returned.

$$x^* = \operatorname*{argmax}_x V(x) \quad (2)$$

One challenge is that even though the candidates $\{x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(K)}\}$ can be constrained to be different strings, it is difficult to enforce semantic diversity. For example, if all the candidates are paraphrases of the same intent, then the choice makes little difference to the outcome of the dialogue. In order to improve the diversity of candidate generation, we take a hierarchical approach of first sampling K unique latent intents $\{z_t^{(1)}, z_t^{(2)}, \ldots, z_t^{(K)}\}$ from $\hat{p}_z(z_t|\mathcal{A}, v, x_{0:t-1})$. Then, for each $z_t^{(i)}$, we choose a candidate turn conditioned on that state:

$$x_t^{(i)} = \operatorname*{argmax}_x p_x(x|z_t^{(i)}, x_{0:t-1})$$

We then estimate the reward of the candidate message using Equation 1, and finally choose a message as in Equation 2.

9. Experiments

9.1. Training Details

We used the following hyper-parameters:: embeddings and hidden states have 256 dimensions; for each unique agreement space $\mathcal{A}$ we learn 50 discrete latent message representations. During training, we optimize the parameters using RMSProp (Tieleman & Hinton, 2012) with initial learning rate 0.0005 and momentum $\mu = 0.1$, clipping of gradients whose L^2 norm exceeds 1. We train the models for 15 epochs with mini-batch size of 16. We then pick the best snapshot according to validation perplexity and anneal the learning rate by a factor of 5 each epoch. For RL, we use a smaller learning rate of 0.0001, and a discount factor γ of 0.95. For supervised learning we tuned based on validation perplexity; for RL we measured the average reward in self-play.

9.2. Baselines

We compare the following models:

- RNN A simple word-by-word approach to generation, similar to Lewis et al. (2017).
- HIERARCHICAL Baseline model in which the two levels of RNN are connected directly, with no discrete bottleneck (§4), similarly to Serban et al. (2016b).
- BASELINE CLUSTERS Our model (Figure 3b) without pretraining the sentence encoder. A latent representation z_t of message x_t is inferred to maximize the likelihood of $p(x_{t+1}, a_t|\mathcal{A}, z_t, z_{0:t-1})p(z_t|\mathcal{A}, x_t)$. This model is closely related to the LATENT INTENTS DIALOGUE MODEL (Wen et al., 2017).
- FULL Our full model, where we first pre-train sentence representations z_t^* to maximize the likelihood $p(x_{t+1}, a_t|\mathcal{A}, z_t^*, z_{0:t-1})p_z(z_t^*|\mathcal{A}, x_t)$, and then we train models to predict $p_x(x_t|z_t^*, x_{0:t-1})$ and $\hat{p}_z(z_t|\mathcal{A}, v, x_{0:t-1})$.

To focus the evaluation on the linguistic and strategic aspects of the dialogue, all systems use the same model for predicting the final agreement represented by the dialogue, which is implemented as a bidirectional GRU with attention over the words of the dialogue.

9.3. Likelihood Models

First, we experiment with models using no RL or rollouts.

9.3.1. PERPLEXITY

Models were developed to maximize the likelihood of human dialogues, which is an indicator of how human-like the language is (we observed qualitatively that the two were strongly correlated). Results are shown in Table 1.

The use of a hierarchical RNN model improves performance over a strong baseline from previous work.

Model	Validation Perplexity	Test Perplexity
RNN	5.62	5.47
HIERARCHICAL	5.37	5.21
BASELINE CLUSTERS	5.61	5.46
FULL	5.37	5.24

Table 1: Likelihood of human dialogues using different models. Our model with discrete message representations is able to achieve state-of-the-art performance, showing that the representations capture relevant aspects of messages for predicting the future dialogue. The size of 95% CI is within 0.03 for each entry.

Model	Score vs. RNN	Score vs. HIERARCHICAL
RNN	5.33	5.17
HIERARCHICAL	5.37	5.08
BASELINE CLUSTERS	4.68	4.66
FULL	6.75	6.57

Table 2: Comparison of different models based on their end-task reward. Our clusters substantially improve reward, indicating that they make it easier for supervised learning to model strategic decision making. The size of 95% CI is within 0.14 for each entry.

Perhaps surprisingly, our hierarchical latent-variable model is also able to achieve state-of-the-art performance. This shows our model's discrete encodings of messages are as informative for predicting the future dialogue as the more-expressive embeddings used by the hierarchical baseline.

9.3.2. COHERENCE OF CLUSTERS

Table 4 shows random samples of messages generated by different clusters from our predicted state model, and the BASELINE CLUSTERS model.

Qualitatively, the states from our model show a higher degree of semantic coherence, and higher linguistic variability. Compared to the BASELINE CLUSTERS , our approach tends to generate more dissimilar surface strings, but with more similar semantics. Our clusters appear to capture *meaning* rather than *form*.

9.3.3. END TASK PERFORMANCE

We measure the performance of the different models on their end-task reward over 1000 negotiations in self-play. Results are shown in Table 2. We find that the use of our latent representations leads to a large improvement in the reward, indicating that our representations make it easier for the supervised model to learn the latent decision making process in the human dialogues it was trained on.

Rollout Type	Score vs. NO ROLLOUTS	Score vs. BASELINE ROLLOUTS
NO ROLLOUTS	5.08	4.91
BASELINE	7.81	6.57
DIVERSE	8.41	7.36

Table 3: Comparison of different rollout strategies for the FULL . DIVERSE rollouts use distinct latent variables to create more semantic diversity in rollout candidates, significantly improving performance. The size of 95% CI is within 0.19 for each entry.

9.4. Hierarchical Planning

Next, we evaluate different rollout strategies:

- BASELINE ROLLOUTS following Lewis et al. (2017), where first K candidate sentences are sampled from the model, and then tokens are sampled iteratively from p_x until reaching the end of the dialogue.
- DIVERSE ROLLOUTS where we first choose the top K unique z_t from $\hat{p}_z$. By choosing unique z_t we aim to increase the semantic diversity of the candidates.

We evaluate compared to the baseline model and word-level rollouts and record the average score. Results are shown in Table 3, and that the DIVERSE ROLLOUTS that use our message representations lead to a large improvement over previous approaches.

9.5. Finetuning with Reinforcement Learning

A challenge in using RL for end-to-end text generation models is that optimising for reward can adversely affect language generation. In selfplay, the model can learn to achieve a high reward by finding uninterpretable sequences of tokens that the baseline model was not exposed to at training time. We compare several RL approaches:

- ALL-RL Reinforcement learning after pre-training with supervised learning.
- ALL-RL+SV Interleaved RL and supervised learning updates, weighting supervised updates with a hyper-parameter α, similarly to Lewis et al. (2017).
- PRED-RL Reinforcement learning only to fine-tune the intent model $\hat{p}_z$, with all other parameters fixed.

We measure both the average reward of the model (a measure of its ability to achieve its goals) and the perplexity of

Cluster	BASELINE CLUSTERS	FULL
1	i can give you the books but , i would need the hat and the balls	i would like the hat and 1 book
	i can do that . i need both balls and one book	i can't give up the hat , but i can offer you the book and 2 balls
2	i need both books and the hat	i want the hat
	how about you get the hat and 1 ball	i need the hat . you can have all the books and the balls
3	i can not make that deal . i need the hat and one book	i can give you the hat and 1 ball
	i can give you the hat and 1 ball	i would like the books and a ball
4	i need two books and the hat	i need the books and the hat
	i need the hat , you can have the rest	i can give you the balls but i need the hat and books
5	i can give you the hat if i can have the rest	could i have the books and a ball ?
	i want one of each	i would like the books and one ball

Table 4: Messages sampled from different clusters, where **2 books**, **1 hat**, and **2 balls** are available. Our method's clusters are much more semantically coherent than the baseline, and correspond to different ways of proposing the same deal.

Model	Score vs. HUMAN	Language quality	Number of turns
FULL + ROLLOUT	7.45	3.55	4.89
RNN + ROLLOUT	6.99	3.43	4.38
FULL + RL	6.26	3.60	6.52
RNN + RL	6.01	3.52	3.99
FULL	5.42	3.68	3.07
RNN	5.30	3.56	3.96
HUMAN	6.64	3.85	6.36

Table 5: Performance of our FULL model and the highly optimized RNN model against humans. In all cases, our FULL model achieves both higher scores and uses higher quality language than RNN .

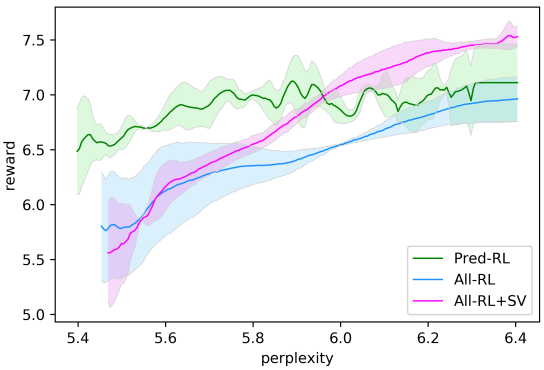


Figure 4: Plotting reward against language quality (lower perplexity is better) during reinforcement learning training, in dialogues with the HIERARCHICAL model. Our method (green) achieves higher rewards while maintaining human-like language (top left of graph).

the model on human dialogues (a measure of how human-like the language is). After hyper-parameter search, we plot the reward of the best model whose perplexity is at most a.

Results are shown in Figure 4. Using RL on all parameters allows high rewards at the price of poor quality language. Only fine-tuning $\hat{p}_z$ allows the model to improve its strategic decision making, while retaining human-like language.

9.6. Human Evaluation

To confirm our empirical results, we evaluate our model in dialogues with people. We ran 1415 dialogues on MTurk, where humans were randomly paired with either one of the models or another human. We then asked humans to rate the language quality of their partner (from 1 to 5). Results are shown in Table 5. We observe that our model consistently outperforms the baseline model (Lewis et al., 2017) both in the end-task reward and the language quality.

10. Analysis

Results in section 9 show quantitatively that our hierarchical model improves the likelihood of human generated language and the average score achieved by the agent. Here, we investigate specific issues that the model improved on, and identify remaining challenges. We analyzed 1000 dialogues between our FULL and the HIERARCHICAL baseline. These models achieve similar perplexity on human dialogues (Table 1).

10.1. Linguistic Diversity

First, we measure the diversity of the agents' language.

RNN language models are known to prefer overly generic messages. In our task, this often manifests itself as short messages such as *deal* or *ok*. We measure the frequency of

simple variations on these messages, and find that the HIERARCHICAL model uses generic messages far more often than FULL (815 times vs. 245).

The messages sent by FULL are also longer on average (8.9 words vs. 6.7, ignoring the end-of-dialogue token), giving further evidence of greater complexity.

We also find that the FULL is substantially more creative in generating new messages beyond those seen in its training data. In total, FULL sends 875 unique message strings, of which 525 (60%) do not appear in the training data. In contrast, HIERARCHICAL sends fewer unique message strings (751), and just 18% of these are not copied from the training data.

10.2. Self-consistency of Messages

Models can output inconsistent messages, such as *I really need the hat. I can give you the hat and one ball*. We searched for messages that mentioned the same item type multiple times, and then manually evaluated whether it was consistent. The FULL model was more prone to this error than HIERARCHICAL (23 times vs. 11), though this fact may be a consequence of its greater creativity, and the problem only occurred in roughly 1% of messages.

10.3. Consistency with Input

We also investigate whether messages are consistent with the context—for example, models may emit messages such as *I'd like the hat and books; you keep the 3 balls* when there are not 3 balls available. We use simple pattern matching for several such errors, and found that the FULL performed slightly better (15 errors vs. 19).

10.4. Consistency with Dialogue Context

Lewis et al. (2017) describe cases where an agent indicates it is simply re-stating an agreement, when it is actually proposing a new deal (e.g. *you get 2 hats / Okay deal, so I get 3 hats*). Interestingly, we found this behaviour only happened with the models using rollouts. While this tactic is effective against our models, it would be frustrating for humans, and future work should address this issue.

10.5. Repetitiveness

Previous work noted that reinforcement learning models were prone to an extortion tactic of simply repeating the same demand until acceptance. We measured how often agents repeated the same message in a dialogue, comparing the ALL-RL+SV model based on previous work, with our PRED-RL model. Our model was substantially less repetitive: only 1% of dialogues contained a repetition of the same message, compared to 12% for the baseline.

Input		
DIVERSE ROLLOUTS	1x**book** *value*=9 1x**hat** *value*=1 4x**ball** *value*=0	
BASELINE ROLLOUTS	1x**book** *value*=0 1x**hat** *value*=6 4x**ball** *value*=1	
DIVERSE ROLLOUTS	I will take the book and hat and you can have the balls.	
BASELINE ROLLOUTS	I need the hat and two balls	
DIVERSE ROLLOUTS	The balls are worthless, I need the hat and the book.	
BASELINE ROLLOUTS	I need the hat or no deal	
DIVERSE ROLLOUTS	Then no deal.	
BASELINE ROLLOUTS	What about the balls?	
DIVERSE ROLLOUTS	You can have the hat but I need the book.	
BASELINE ROLLOUTS	How about I get the balls and 1 hat?	
DIVERSE ROLLOUTS	Ok	
Output		**Reward**
DIVERSE ROLLOUTS	1x**book**	9/10
BASELINE ROLLOUTS	1x**hat** 4x**ball**	10/10

Figure 5: Dialogue between two models using different types of rollouts. The DIVERSE ROLLOUTS model makes several attempts to win the *hat*, before compromizing.

11. Related Work

Traditional goal-orientated dialogue models have first generated symbolic *intents*, capturing the meaning of the message, and then generated text to match the intent (e.g. Williams & Young (2007), Keizer et al. (2017)). Our approach can be seen as a latent model for generating intents. Our model is most closely related to other recent latent variable hierarchical dialogue models from Serban et al. (2016c), Wen et al. (2017) and Cao & Clark (2017). An important difference is that both these approaches optimize latent representations z to maximize the likelihood of generating the next message—whereas our model pretrain's z to maximize the likelihood of the continuation of the dialogue, to better capture the semantics of the message rather than its surface form. While other ways of learning discrete latent representations were proposed recently (van den Oord et al., 2017; Kaiser & Bengio, 2018), we have shown that our approach leads to higher performance on a strategic dialogue task.

Other work has explored generating sentence embeddings for open domain text—for example, based on maximizing the likelihood of surrounding sentences (Kiros et al., 2015), supervised entailment data (Conneau et al., 2017), and auto-encoders (Bowman et al., 2015).

12. Conclusion

We have introduced a novel approach to creating sentence representations, within the context of an end-to-end strategic dialogue system, and have shown that our hierarchical approach improves text generation and planning. We identified a number of challenges faced by previous work, and show empirically that our model improves on these aspects. Future work should apply our model to other dialogue settings, such as cooperative strategic dialogue games (He et al., 2017), or multi-sentence generation tasks, such as long document language modelling (Merity et al., 2016).

References

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Józefowicz, R., and Bengio, S. Generating sentences from a continuous space. *CoRR*, abs/1511.06349, 2015.

Cao, K. and Clark, S. Latent variable dialogue models and their diversity. *CoRR*, abs/1702.05962, 2017.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.

Das, A., Kottur, S., Moura, J. M., Lee, S., and Batra, D. Learning cooperative visual dialog agents with deep reinforcement learning. *arXiv preprint arXiv:1703.06585*, 2017.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.

He, H., Balakrishnan, A., Eric, M., and Liang, P. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In *Association for Computational Linguistics (ACL)*, 2017.

Kaiser, L. and Bengio, S. Discrete autoencoders for sequence models. *CoRR*, 2018.

Keizer, S., Guhe, M., Cuayhuitl, H., Efstathiou, I., Engelbrecht, K.-P., Dobre, M., Lascarides, A., and Lemon, O. Evaluating persuasion strategies and deep reinforcement learning methods for negotiation dialogue agents. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, 2 2017.

Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R. S., Torralba, A., Urtasun, R., and Fidler, S. Skip-thought vectors. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pp. 3294–3302, Cambridge, MA, USA, 2015. MIT Press.

Lewis, M., Yarats, D., Dauphin, Y. N., Parikh, D., and Batra, D. Deal or no deal? end-to-end learning for negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language*, pp. 2433–2443. Association for Computational Linguistics, September 2017.

Li, J., Galley, M., Brockett, C., Gao, J., and Dolan, B. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*, 2015.

Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J., and Jurafsky, D. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Serban, I. V., II, A. G. O., Pineau, J., and Courville, A. C. Piecewise latent variables for neural variational text processing. *CoRR*, abs/1612.00377, 2016a.

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A. C., and Pineau, J. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, pp. 3776–3784, 2016b.

Serban, I. V., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A. C., and Bengio, Y. A hierarchical latent variable encoder-decoder model for generating dialogues. *CoRR*, abs/1605.06069, 2016c.

Tieleman, T. and Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

van den Oord, A., Vinyals, O., and Kavukcuoglu, K. Neural discrete representation learning. *CoRR*, pp. 6306–6315, 2017.

Wen, T., Miao, Y., Blunsom, P., and Young, S. J. Latent intention dialogue models. *CoRR*, abs/1705.10229, 2017.

Williams, J., Raux, A., Ramachandran, D., and Black, A. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pp. 404–413, 2013.

Williams, J. D. and Young, S. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422, 2007.